INFORME DE CALIDAD (SPRINT 2)

AUTOR: BÁRBARA MARTÍNEZ CARCEDO Y MIGUEL CASAMICHANA

ANÁLISIS 14 NOVIEMBRE 2020

CAPTURA

INCIDENCIAS

El análisis no pasa los criterios de calidad de la organización debido a que la calificación de confiabilidad (reliability) es B, cuando debería ser A. Esto es debido a un único bug presente en la clase Vehículo. Además de esto, la deuda técnica (technical doubt) es de 7h 33min, cuando debería de ser inferior a 4h 10min.

Por otro lado, encontramos 74 issues de mantenibilidad (code smells) encontrándose la mayoría en MisVehiculosActivity.java (13) y MainActivity.java (20). En cuanto a la severidad de estos code smells, encontramos de todo tipo siendo las más abundante las de tipo info(24) y minor(32) , aunque también encontramos del resto de tipos, como ya he dicho, 2 blocker, 2 critical, 15 major.

PLAN DE ACCIÓN

1. Arreglar el bug de la clase Vehículo.
2. Eliminar los issues de severidad major, alojados la mayoría en la clase PresenterGasolineraTest.
3. Solucionar el siguiente error “Make xxx a static final constant or non-public and provide accessors if needed” que supone 1h 30min de deuda técnica, y aparece en las clases MainActivity, MisVehiculosActivity y FormActivity.
4. Eliminar los issues con severidad blocker, en los que simplemente hay que añadir algún test a la clases VerDetalleUITest y GasolineraTest.
5. Eliminar los issues con severidad critical dados por los siguientes errores “*Rename this constant name to match the regular expression '^[A-Z][A-Z0-9]*(_[A-Z0-9]+)*$'*” en la clase IFiltro y “*Define a constant instead of duplicating this literal "Campo Requerido" 3 times* en la clase FormActivity.
6. Eliminar los imports que no se usen, o que estén duplicados.
7. Eliminar la líneas de código comentadas.

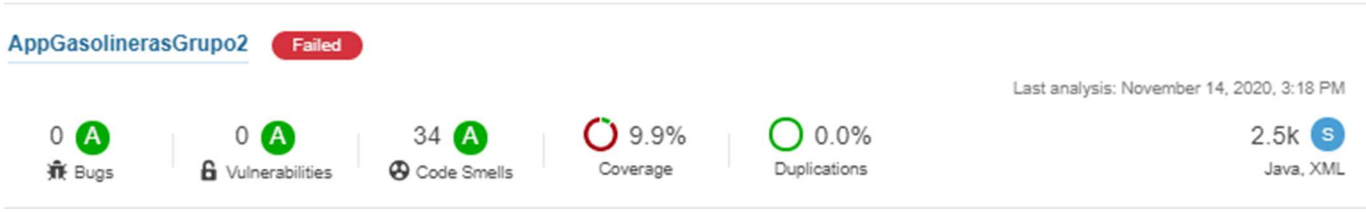
-Comentarios:

Con estas medidas conseguiríamos rebajar la deuda técnica 4h 58 min, teniendo en cuenta los errores por desuso (depricated) de algunos métodos, esto supone 2h 30min, pero tras comentárselo al profesor de la asignatura me comunicó que no tuviera en cuenta estos errores para cumplir con los requisitos mínimos de calidad. Dicho esto, sin tener en cuenta estos errores la deuda técnica sería de 4h 58min – 2h 30min = 2h 28min, por lo que sí que se cumplirían el requisito de tener una deuda técnica inferior a 4h 10min.

Autor: Bárbara Martínez Carcedo

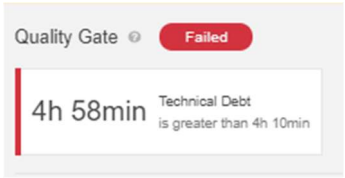
ANÁLISIS 14 NOVIEMBRE 2020

CAPTURA

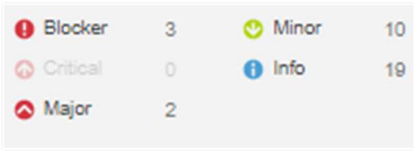


INCIDENCIAS

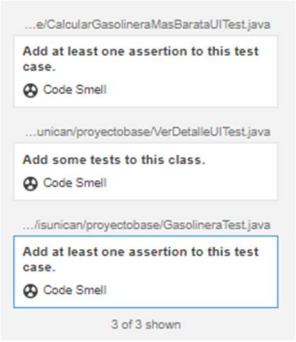
A primeras se puede apreciar que pese a que todos los criterios tengan una calificación de A, no se pasan los criterios de calidad. Esto se debe a una deuda técnica superior a 4 horas y 10 minutos, en concreto la que se muestra a continuación:



Se pueden apreciar 34 code smells que suman las casi 5 horas de deuda técnica mencionada de entre los cuales se distinguen 3 de severidad *Blocker,* 2 de severidad *Major*, 10 de severidad *Minor* y 19 de severidad *Info*.



En cuanto a los code smells de severidad Blocker se encuentran en las clases de Test principalmente debido a que en algunas de éstas, sus test cases no tienen asserts debido a que éstos (test cases) están automatizados con Expresso o que simplemente están inacabados.



Además, el mayor número de code smells se encuentran en las clases “MainActivity”, “MisVehiculosActivity” y “Gasolinera” en las cuales se encuentran 9, 7 y 5 code smells.

PLAN DE ACCIÓN

1. Solucionar los 3 *code smell* con severidad *Blocker* (25 minutos) presentes en las clases de test mencionadas: “CalcularGasolineraMasBarataUITest.java”, “VerDetalleUITest.java” y “GasolineraTest.java”.
    a. El primero corresponde a la clase “CalcularGasolineraMasBarataUITest” e indica “Add at least one assertion to this test case”. Como bien mencioné anteriormente, se debe a que el test case sobre el que se especifica este code smell está comentado ya que está inacabado. Bastaría con añadir un assert, como Assert.assertTrue(true) hasta que se finalice el test.
    b. El segundo corresponde a la clase “VerDetalleUITest” e indica “Add some tests to this class”. Bastaría con añadir un test case simple hasta que se finalice la clase finalmente.
    c. El tercero, y último, code smell de severidad *Blocker* corresponde a la clase “GasolineraTest” e indica “Add at least one assertion to this test case”. Tenemos al igual que antes, otro test case comentado completamente, por lo que con añadir un assert simple valdría hasta finalizar el test case correctamente.
2. Solucionar el code smell de severidad *Major* (5 minutos) de la clase “FormActivity” el cual indica “Remove this unused "presenterVehiculos" private field” el cual consiste simplemente en borrar un campo que no es usado.
3. Solucionar el code smell de severidad *Major* (5 minutos) de la clase “Vehículo” el cual indica “Merge this if statement with the enclosing one” el cual consiste simplemente en introducir una condición de un if dentro de otro que anteriormente le englobaba. Tras esto nos encontraríamos solo con fallos de severidad *Minor* e *Info*.
4. Dentro de la clase “Gasolinera” se pueden arreglar 3 issues de severidad *Minor* (6 minutos). Dos de ellos son referentes al campo “DEPOSITO”: “Rename this field "DEPOSITO" to match the regular expression '^[a-z][a-zA-Z0-9]*$'” y “Make this final field static too” por los cuales con cambiar el nombre como se especifica y hacer el atributo estático sería suficiente; y otro es eliminar un import en desuso.
5. Dentro de la clase “FormActivity” se pueden arreglar otros 2 issues de severidad *Minor* (10 minutos) los cuales indican que se borre el atributo de “presenterVehiculos” y que se declare, si hicese falta, como variable local dentro de los métodos en los que se utilice.
6. Dentro de la clase “MisVehiculosActivity” se encuentra otro issue de severidad *Minor* (2 minutos) el cual indica “Remove this empty statement” el cual consiste en borrar una línea de código vacía.

-Comentarios

Cabe destacar que los issues precedidos por usos de Thread.sleep() son justificados debido a la necesidad de éstos para cargar elementos de las interfaces o de la aplicación. También, han sido ignorados los issues del tipo “… is deprecated”. Todos éstos suman una deuda técnica bastante alta.

Autor: Miguel Casamichana Bolado

ANÁLISIS 16 NOVIEMBRE 2020

CAPTURA

El análisis no pasa los criterios de calidad de la organización debido a que la la deuda técnica (technical doubt) es de 4h 27min, cuando debería de ser inferior a 4h 10min.

Por otro lado, encontramos 31 issues de mantenibilidad (code smells). Esto code smells presentan las siguientes severidades, info(19), minor(7), major(4) y critical(1).

PLAN DE ACCIÓN

1. Arreglar el único code smells con severidad critical de la clase FormActivity.
2. Arreglar los code smells con severidad major.
3. Eliminar los casteos innecesarios de Gasolinera en ConDescuentoFiltro.java y SinDescuentoFiltro.java
4. Solucionar el siguiente error en la clase PresenterVehiculos “Replace the type specification in this constructor call with the diamong operator(“<>”)”.

-Comentarios

1. Con estas modificaciones ya se deberían cumplir los criterios de calidad mínimos.

Autor: Bárbara Martínez Carcedo

ANÁLISIS 17 NOVIEMBRE 2020

CAPTURA

INCIDENCIAS

El análisis no pasa los criterios de calidad de la organización debido a que la calificación de confiabilidad (reliability) es C, cuando debería ser A. Esto es debido a un único bug presente en la clase Vehículo.

Por otro lado, encontramos 19 issues de mantenibilidad (code smells) En cuanto a la severidad de estos code smells, tenemos de tipo info(16) y minor(2) y major(1).

PLAN DE ACCIÓN

1. Arreglar el bug de la clase Vehículo.
2. Resolver el code smell con severidad major, en realidad hay 2 code smells con esta severidad pero uno de ellos es el bug.

-Comentarios

1. Solo con llevar a cabo el punto 1 del plan de acción se cumplirían los requisitos mínimos de calidad, pero llevo a cabo el punto 2 también para mejorar la calidad.

Autor: Bárbara Martínez Carcedo

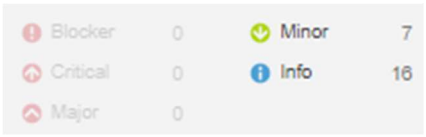
ANÁLISIS 17 NOVIEMBRE 2020

CAPTURA

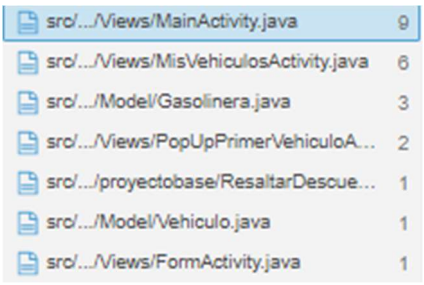
INCIDENCIAS

Se puede apreciar a simple vista que se cumplen todos los criterios de calidad: no se supera una deuda técnica superior a 4h y 10 minutos, no hay bugs y tampoco vulnerabilidades.

Sin embargo, sigue habiendo presencia de un número elevado de code smells, en concreto, 23 code smells. 7 de éstos son de severidad *Minor* y 16 son de severidad *Info*.



Además, el mayor número de code smells se encuentra en la clase “MainActivity.java” en la cual hay 9 de 23 code smells.



PLAN DE ACCIÓN

1. Arreglar los dos code smell (2 min. cada uno) de severidad Minor que citan lo siguiente“Replace this if-then-else statement by a single method invocation” dentro de la clase “Gasolinera.java” el cual consiste en simplificar un bloque de código if-else.
2. Arreglar el code smell de “Gasolinera.java” (2 min.) en el cual se cita lo siguiente “Reorder the modifiers to comply with the Java Language Specification” y consiste en reorganizar el orden de los modificadores del atributo “DEPOSITO”.
3. Arreglar el code smell de la clase “MainActivity.java” (5 min.) en el cual se cita lo siguiente “Combine this catch with the one at line 385, which has the same body”. Bastaría con juntar dos catch ya que tienen el mismo cuerpo.
4. Arreglar el code smell de severidad *Minor* de la clase “Vehículo.java” (2 min.) en el cual se cita lo siguiente “Replace this if-then-else statement by a single return statement”. Bastaría con simplificar el bloque de código mencionado.

COMENTARIOS

Cabe destacar que los issues precedidos por usos de Thread.sleep() son justificados debido a la necesidad de éstos para cargar elementos de las interfaces o de la aplicación. También, han sido

ignorados los issues del tipo "… is deprecated". Todos éstos suman una deuda técnica bastante alta. Además, cabe destacar que debido a planes de acción y diversos análisis de calidad previos la deuda técnica se ha reducido bastante al igual que el número de code smells a tratar.

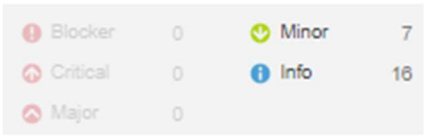
ANÁLISIS 18 NOVIEMBRE 2020

CAPTURA

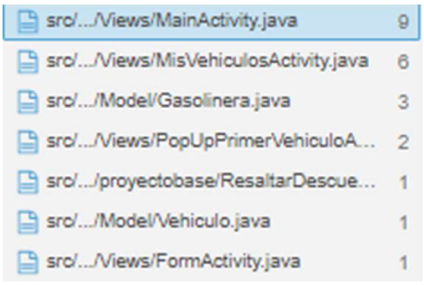
INCIDENCIAS

Se puede apreciar a simple vista que se cumplen todos los criterios de calidad: no se supera una deuda técnica superior a 4h y 10 minutos, no hay bugs y tampoco vulnerabilidades.

Sin embargo, sigue habiendo presencia de un número elevado de code smells, en concreto, 23 code smells. 7 de éstos son de severidad *Minor* y 16 son de severidad *Info*.



Además, el mayor número de code smells se encuentra en la clase "MainActivity.java" en la cual hay 9 de 23 code smells.



PLAN DE ACCIÓN

1. Arreglar los dos code smell (2 min. cada uno) de severidad Minor que citan lo siguiente"Replace this if-then-else statement by a single method invocation" dentro de la clase "Gasolinera.java" el cual consiste en simplificar un bloque de código if-else.
2. Arreglar el code smell de "Gasolinera.java" (2 min.) en el cual se cita lo siguiente "Reorder the modifiers to comply with the Java Language Specification" y consiste en reorganizar el orden de los modificadores del atributo "DEPOSITO".
3. Arreglar el code smell de la clase "MainActivity.java" (5 min.) en el cual se cita lo siguiente "Combine this catch with the one at line 385, which has the same body". Bastaría con juntar dos catch ya que tienen el mismo cuerpo.

4. Arreglar el code smell de severidad *Minor* de la clase “Vehículo.java” (2 min.) en el cual se cita lo siguiente “Replace this if-then-else statement by a single return statement”. Bastaría con simplificar el bloque de código mencionado.

COMENTARIOS

Cabe destacar que los issues precedidos por usos de Thread.sleep() son justificados debido a la necesidad de éstos para cargar elementos de las interfaces o de la aplicación. También, han sido ignorados los issues del tipo “… is deprecated”. Todos éstos suman una deuda técnica bastante alta. Además, cabe destacar que debido a planes de acción y diversos análisis de calidad previos la deuda técnica se ha reducido bastante al igual que el número de code smells a tratar.

Autor: Miguel Casamichana Bolado

ANÁLISIS 19 NOVIEMBRE 2020

CAPTURA

El análisis si que pasa los criterios de calidad si no tenemos en cuentas la deuda técnica (technical doubt) que nos aporta el uso de métodos deprecated, 2h 43 min. La deuda técnica que aparece en el sonarcloud es de 5h 53 min pero, si le restamos la deuda técnica por los métodos que están depricated, nos quedaría una deuda técnica de 3h 10min, por debajo de 4h 10min por lo que pasa los criterios de calidad mínimos.

Por otro lado, encontramos 31 code smells, con severidad critical(6), major(2), minor(3) e info(20), y 8 security hotspot.

PLAN DE ACCIÓN

1. Corregir los 8 security hotspot.
2. Solucionar los 6 code smells con severidad crítica.
3. Solucionar los 2 code smells con severidad major.

-Comentarios

1. El profesor de la asignatura me comentó que no tuviéramos en cuenta por el momento la deuda técnica aportado por los métodos que están en desuso (depricated).

Autor: Bárbara Martínez Carcedo